ITM Web of Conferences **80**, 02001 (2025) https://doi.org/10.1051/itmconf/20258002001
ACAAI 2025

Variance and robustness aware UCB variants for evaluating e-commerce fulfilments

Mingguangyi Yang[1]*
[1]New York University, Tandon School of Engineering, Department of Mathematics, Brooklyn, NY, USA

Abstract. The online allocations of inventory to different product categories are naturally a multi-armed bandit problem. Companies need efficient and low-regret exploration to allocate the deep inventories to optimal options. This research intends to do an offline evaluation of UCB variants on a public e-commerce sales dataset, setting the arms as product categories and binary rewards from the terminal fulfilment status. It is comparing UCB1, Asymptotically Optimal UCB, UCB-Variance, Bootstrapped UCB, and IQR-UCB algorithms for performances in regret, pull allocation, and runtime comparison. The UCB-Variance algorithms result in the optimal performance combination with low regret and baseline runtime, while optimal-UCB offers a slight improvement and the baseline runtime. Bootstrapped UCB offers little gain at high costs, while the IQR-UCB shows overly aggressive exploration on binary rewards due to the vanishing bonus. For the category-level allocation, the UCB-Variance algorithm seems to be the most cost-benefit, and Optimal-UCB is a safe anytime baseline. However, conservative success labelling and less operational results provides rooms for further optimization. We should also evaluate Bootstrapped UCB with upper-quantile and lazy updates, IQR-UCB with alternative scales and adding floors, and non-stationary and contextual settings to better examine the full potentials of these methods.

1 Introduction

E-Commerce gradually became the prevalent lifestyle of modern citizens, especially after the pandemic quarantine that propelled everyone to stay indoors. Among the purchases on the platforms, the essential point is the satisfaction of the customers, while cancellation or return serving as the mirror image of attitude. Research states that online purchases, especially on fits, are more prone to high return rates since the customer cannot physically evaluate the looks on them during the purchases [1]. When we focus solely on the purchasing process, the studies have indicated about 88% global cart abandonment rate after they placed the items in the carts [2]. It further underscores the sensitivity of shoppers towards the item quality as well as the fulfilment friction. Each of the unsatisfied pick would result in doubly the cost for the platform – not only the revenue loss from freight and repackaging, but also the reputational damage that will harm the future repetitive purchases. The same customer chain

*Corresponding author: My2553@nyu.edu

research evaluated the single return cost to 17% of the overall prime costs of order, consisting mainly of freight and handling expenses [1].

To address this issue, it is essential to choose and develop the right algorithm for real-time decision making, searching the best solutions while minimizing the evaluation costs. Although the standard A/B testing utilizes the exploration and exploitation steps to evaluate the best options, it would inevitably waste significant impressions on the low-satisfaction category testing and suffer even greater from seasonal shift in purchase habits since it is rigid to readjust to the new trends. A recent research article evaluates the comparison between A/B testing and Multi-armed Bandit (MAB) algorithms and asserts that: A/B testing must spend a fixed period of the poor performing experience, resulting in the bad business performances, while in contrast MAB resolves this issue by dynamically deciding the best options [3]. Thus, as an alternative, we plan to experiment the MAB algorithm, specifically the UCB variants, as the analytic lens for the problem. By testing and shrinking the confidence intervals on a cleaned e-commerce sales dataset, we plan to evaluate and compare these variants' performances in deciding the highest satisfactory item category in purchases. By benchmarking these four upper confidences bound variants, we demonstrate that replacing mean-only uncertainty with variance-, bootstrap-, and especially IQR based bonuses strategies would let the online retails discover the highest satisfaction categories in a more accurate and efficient way, avoiding tens of thousands in reverse logistics costs.

2 Related Work

The most standard Upper Confidence Bound Algorithm sets an index for each arm while exploring the best options. Aside from determining the options returning the highest reward, it also adds a safety cushion in the reward intervals to account for the uncertainties. This ensures the minimal mistakes in choosing the arm and keeps an efficient pace of testing. Peter Auer, from the University of Technology Graz in Austria, has found to use UCB1 to achieve logarithmic regret on the sample without requiring a prior understanding of the reward distribution of the options [4]. It asserts that the cushions help to keep the total mistake cost growing only slowly over time while maintaining its advantages in searching the best option. The later authors, based on the standard UCB, developed a noise check in the algorithm to account for reward fluctuations in the arms over time. Audibert-Munos-Szepesvári published a paper in 2009 that incorporate the "empirical variance" of different arms into the considerations of the exploration algorithm. The results of the noisy-aware rule have come out to surpass greatly the original version and even faster early learning when test groups differ in stability [5]. This has provided a solid support for considering the noise in the UCB algorithm, especially for a greatly swinging dataset.

In addition to the optimizations in the bonuses, researchers have implemented renovations on the sample mean calculation in the UCB index. Using bootstrapping, a remixing of past sample history to recalculate the means, it provides a data-driven cushion in the exploration instead of the simple math-formula cushion. When reward distributions are skewed, it would be useful and avoid the negative influence of these irregular patterns. The examination of this novel method shows that this remix avoids the over testing of obviously poor items and "reduces regret relative to Vanilla-UCB" under the heavy-tailed reward distribution [6]. It is still a visible optimization with lower cumulative regret under the heavy-tailed noises. There have also been proposals regarding building the cushion from direct sections of the data. While many bandits are trying to make optimization on the mean rewards, Szörényi et al. introduced the idea of "quantile bandits" that cares about reward quantile and maintains optimism on the empirical quantile instead of the mean, which has also achieved a logarithmic regret [7].

All the additional UCB variants has proposed a renovated way of evaluating the best options in the online purchasing merchandise. It is also critical to point out that recent studies have evaluated the effectiveness of the bandit systems mainly through metrics of "click-through rate", "dwell time", or "ad revenue", whereas return rate is not mentioned among the evaluations [8]. It indicates a possible gap that is worthy to fill in the evaluation metrics of the bandit systems. The additional links between the bandit system and the inventory planners can be further reinforced by the research that applies contextual bandits to "warehouse stock decisions". Compared with standard rule baseline by the company, the application cuts the safety-stock holding cost by 12% [9]. It is obviously seen from this research how the bandit system can be used on the dollar saving objectives in the e-commerce inventory, yet it is still lacking evaluations on the post-delivery customer satisfaction, or about the return rate.

3 Problem Setting

In this problem, we will use the context-free multi-armed bandit (MAB) algorithm to enhance the e-commerce fulfilment quality. After the customer makes a purchase for specific product categories on the e-commerce platform, we will track a binary fulfilment outcome of the order and assign a reward for that arm of the bandit system. The objective is to minimize the cumulative regrets of the algorithm choosing the arms over time. Equivalently, we are trying to maximize the expected kept-order rate for the e-commerce platforms to help them avoid suffering from harmful return costs and optimize the product allocation process.

The arms of the bandit system would be set to the top-level product categories after cleaning and merging the dataset. To avoid the issue of sparsity caused by using product SKU, the merchandisers usually do category-level routing for making decisions for its easiness in analysing and managing the actual purchases. For the rewards from the bandit arms, we would use a Bernoulli setting: set reward to 1 if the order status is shipped/delivered, to 0 if status cancelled, returned, or undelivered. To clarify any ambiguity, if the order shows shipped, but later a returned appears for the same order ID, we would follow the last terminal state. In other words, we are concerning about the post-purchase quality, instead of the click rates, dwell time, or other parameters.

Theoretically, with a fixed Bernoulli reward distribution with mean $\mu_k \in [0,1]$ and independence across the rounds with chosen arms (standard bandit assumption), we can apply the stochastic model on the system and assume a bounded reward with sub-Gaussian distributions [10]. It adds additional security to the UCB index bonuses. When the policy picks an arm k at time t, it is essential to consume the next outcome from that arm's time-ordered list in the online split without resampling. This can ensure the within arm assumption of independent and identically distributed, preventing the leakage over time [11].

4 Dataset

The training dataset is from a public data.world platform, about the E-Commerce sales statistics [11]. It has a bundle of 7 CSVs and is widely used in tutorials and analyses. For the bandit system training, we will mainly use the "Amazon Sale Report", as it contains the information of product category and the delivery states of them along with the dates. It has approximately 129,000 rows of sales data for Amazon's apparel section in the second quarter of 2022.

From the table, we will restrict ourselves to the "Order ID" and "Date" for order identification, "Courier Status" to check the order status as rewards, and "Category" to serve as the bandit arms. After sorting the records by their dates, it keeps the last terminal status for each unique Order ID. For the order's Courier Status, we will map it to a Bernoulli reward distribution, 1 if shipped or shipped-delivered to buyer, 0 if cancelled, returned or otherwise. For the dates, we will parse it to a proper timestamp and sort chronologically. Finally, we will also drop any rows with missing "Status" or "Category", leaving approximately 120,000 rows at the end.

5 Algorithms / UCB Variants

To evaluate the best product category option from the e-commerce sales data, we will use the UCB variants of the multi-armed bandit algorithm, as stated above. They operate on the category-level Bernoulli rewards with shared assumptions of stationary, bounded rewards, context-free bandits, and same online stream for fairness. All the variations are index policies with equations (1) and (2).

$$I_k(t) = \hat{\mu}_k + bonus_k(t) \tag{1}$$

$$a_t = \arg\max I_k(t) \tag{2}$$

For each arm k of the bandit system, you will keep a point estimate of the mean reward $\hat{\mu}_k$ and an optimism bonus, the confidence radius, that shrinks as you collect more data. At round t, after you compute the UCB index according to the specific formula, you will pick the arm with the largest index $I_k(t)$. Generally, arms with few pulls will get a bigger bonus so that there will be forced exploration for these neglected arms, and arms with many pulls a much smaller bonus. By the time of exploration, the bandit will finally determine and exploit one arm with the largest UCB index upper bound, which is approximately around its mean reward due to the large sample size. This arm would be the optimal choice from the algorithm.

5.1 UCB1

The UCB1 algorithm, also known as the Hoeffding-UCB, is the baseline of all variants. It performs better than the standard ETC algorithm and lets arms with fewer observation get a larger optimism bonus that will shrink over time, with the UCB index defined as equation (3) [11].

$$I_k(t) = \hat{\mu}_k(t-1) + \sqrt{\frac{4\log n}{T_k(t-1)}} \tag{3}$$

The n represents the horizon, $T_k(t-1)$ is how many times arm k has been pulled before round t. Each suboptimal arm will be pulled $\frac{16\log n}{\Delta_k{}^2}$ times, where $\Delta_k = \mu_{optimal} - \mu_k$. This algorithm requires a prior knowledge of the horizon and should better be used when binary or bounded rewards dominate, and arm variances are similar.

5.2 Asymptotically Optimal UCB

While ordinary UCB1 requires the prior knowledge of horizon and may over explore, asymptotically optimal UCB is an essential improvement, with index defined as equation (4) [11].

$$I_k(t) = \hat{\mu}_k(t-1) + \sqrt{\frac{2\log(1 + t * ln^2 t)}{T_k(t-1)}} \tag{4}$$

In comparison, it does not require knowledge about the horizon and would expand the confidence interval of the index over time to ensure that no arm is missed out, while it also has the sampling counting denominator at the bottom to shrink the interval after sampling. It would select the suboptimal arms $\frac{2\log n}{\Delta_k{}^2}$ times, resulting in a smaller regret term.

5.3 UCB-V

The UCB-Variance algorithm uses an empirical variance to scale the confidence radius. Stable arms with low variance will get a smaller bonus, leading to a faster lock in, while noisy arms will get more exploration. The UCB-V index will be calculated as equation (5) [5].

$$I_k(t) = \hat{\mu}_k(t-1) + \sqrt{\frac{2\widehat{v_k}\log(t+1)}{T_k(t-1)}} + \frac{3\log(t+1)}{T_k(t-1)} \tag{5}$$

The variance term $\widehat{v_k}$ is computed for standard Bernoulli distribution, as $\widehat{v_k} = \hat{\mu}_k(1 - \hat{\mu}_k)$ for each arm. Also, we will start using the variance term when the number of pulls is greater or equal to 3 since early variance estimates are noisy. If a category becomes extremely pure, near 0 or 1, adding a tiny floor makes the exploration never fully dies before it is enough confident. It will therefore provide a better performance when arm variance differs.

5.4 Bootstrapped UCB

For the Bootstrapped UCB algorithm, instead of optimizing on the confidence interval term, we are optimizing the sample mean term. By using the bootstrap mean as the location estimate for each arm and adding the asymptotically optimal UCB bonus, it will keep the location non-parametric while still preserving a relatively small exploration regret. For each arm k, the Bootstrapped-UCB index will be computed as equation (6) [6].

$$I_k(t) = \hat{\mu}_k^{boot}(t) + \sqrt{\frac{2\log(1 + t * ln^2 t)}{T_k(t-1)}} \tag{6}$$

We will keep a history list of observed rewards, and each round resample one bootstrap of size n_k from the history. In other words, we will draw the bootstrap samples with replacement from the history list of rewards and add the bonus to select the arm with the highest index. This is a model-free location estimate, which is tolerant to skews and tailed behaviours than the previous methods. It also takes into considerations the reward uncertainty by the resampling of past rewards.

5.5 IQR-UCB

For the IQR-UCB, it is inspired by the incorporation of quantiles in the UCB algorithm [7]. Using the robust scale of IQR instead of variance to size the confidence radius, it prevents the outliers from blowing up the exploration phase. The IQR-UCB index of this algorithm will be computed as equation (7).

$$I_k(t) = \hat{\mu}_k(t-1) + \alpha * \frac{IQR_k(t-1)}{\sqrt{T_k(t-1)}} \tag{7}$$

We will also keep a past reward list of each arm and compute the interquartile range to be incorporated into the confidence radius. There is also a term α to test controlling the confidence radius through single knob, with a decay of $\frac{1}{\sqrt{T_k(t-1)}}$ to prevent permanent over exploration on stable arms.

6 Experimental Setup

After we have generated a cleaned data frame according to the requirements, we will build a time-ordered queue per category to feed into the algorithms. For the horizon, we will test the sample size of 5000, 50,000, and 100,000 to track the performance shifts of the UCB variants as the horizon differs. For the evaluation metrics, we will evaluate the cumulative regrets of the algorithms $R(T)$ based on this equation (8).

$$R(T) = \sum_{t=1}^{T} (\mu_{optimal} - \mu_a) \tag{8}$$

In other words, we are computing the cumulative loss of selecting the sub-optimal arms over time. Additionally, we will look at the arm selections of the algorithms as well as the runtime analysis to compare the algorithms muti-dimensionally.

7 Results & Discussion

7.1 Cumulative Regret

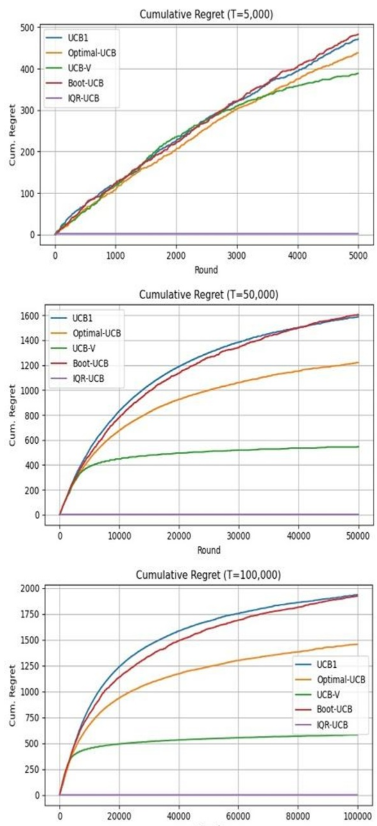

Fig. 1. Cumulative Regret of UCB variants under the horizons of n = 5000, 50000, and 100000

As shown in Fig.1, we can observe that all the variants have similar cumulative regret performances, except for the extremely low regret of IQR-UCB. As the sample size grows, most variants began to show the logarithmic trend in the graph. Take the IQR-UCB aside for later, UCB-V has the flattest shape after the first few thousand rounds and ends at around regret of 580, much lower than the 1933 for UCB1 and 1455 for optimal-UCB. The optimal-UCB is consistently below UCB1 at all horizons, while the gap widens with time. Bootstrapped UCB seems to track the UCB1 mostly, with little gains from its optimization ending up in regret of 1923, and IQR-UCB is nearly flat from the start, showing a sign of extremely aggressive exploration ending up in single-digit regret.

Since optimal-UCB uses a slightly smaller anytime log schedule, it explores a little bit less than UCB1 without needing the horizon, explaining its priority over UCB1. UCB-V scales the bonus by empirical variance and concentrates on genuinely ambiguous arms in the exploration. It is possibly due to the low variance in the amazon sales dataset that eliminates the stable categories quickly and render the exploration much more aggressively that makes UCB-V prevails a lot than most algorithms. The Bootstrapped UCB keeps an analytic bonus but replaces the mean with a single bootstrap mean, possibly adding noise and costs on Bernoulli rewards that elevates its cumulative regret. For the IQR-UCB, as it uses IQR as scale, the bonus may collapse fast to 0 due to the Bernoulli rewards, making the policy extremely greedy and end up exploiting unreasonably fast.

If the objective is low regret with low cost, we should pick UCB-V; if goal is a simple anytime baseline, Optimal-UCB is a safe upgrade over UCB. For the IQR-UCB, we can treat it as a value-metric option, as its near-zero regret on binary rewards can be fragile after unluck starts. Bootstrapped UCB offers no regret win here and does not worth the computation unless switching to a better optimization.

7.2 Arm Selection

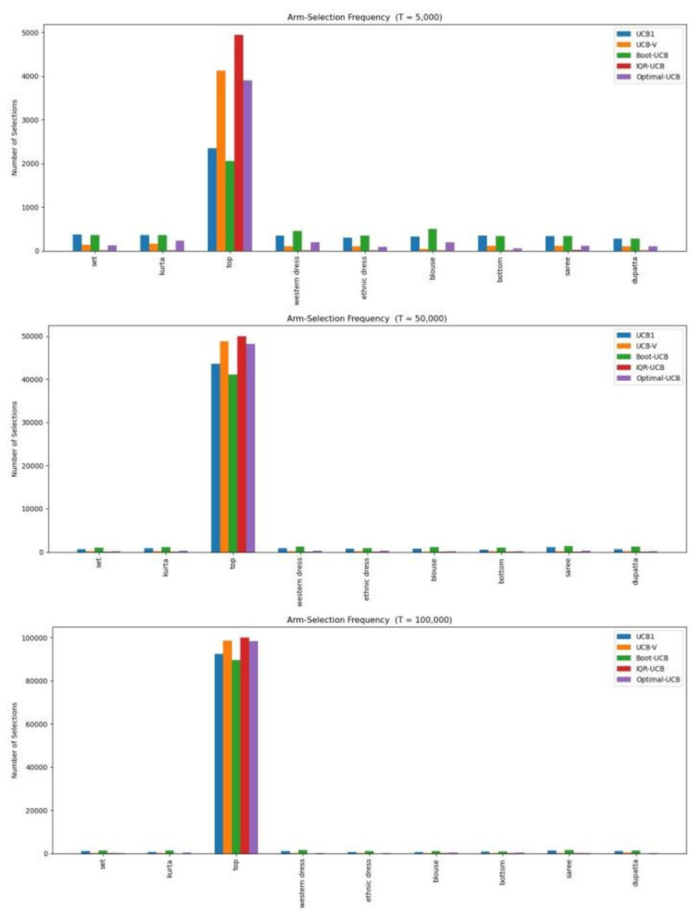

Fig. 2. Arm Selection Frequency of UCB variants under the horizons of n = 5000, 50000, and 100000

As shown in Fig.2, in different horizons, we can see that all algorithms allocate most pulls to the best category. Optimal-UCB has a faster convergence than UCB1 but keeps more residual traffic on weaker traffic than UCB-V. UCB-V reaches nearly monopoly early without concentrating too fast like the IQR-UCB in only a few trials. The Bootstrapped UCB resembles the UCB1 with noticeable mass on the suboptimal categories for longer.

It corresponds to our analysis of the algorithms earlier. IQR-UCB has a tiny bonus on binary rewards, leading to its nearly deterministic choice after a handful of samples, while variance-aware bonuses in UCB-V shrink also rapidly for low variance arms and lead to fast elimination. Optimal-UCB reduces exploration uniformly with smaller radius and hence has a modestly faster convergence than UCB1. The Bootstrapped UCB, instead of reducing the bonus, jitters the location with a bootstrap mean, therefore has similar allocation dynamics like UCB1.

Therefore, the customer-exposure cost can be minimized sooner and nicely with UCB-V, becoming useful when testing hurts users and incurs many costs. For monitoring all the algorithms, it is also useful to track the pull-share over time to avoid under-exploration risks, and we may also enforce safety floors in the formula to avoid the possible greedy exploration.

7.3 Runtime Comparison

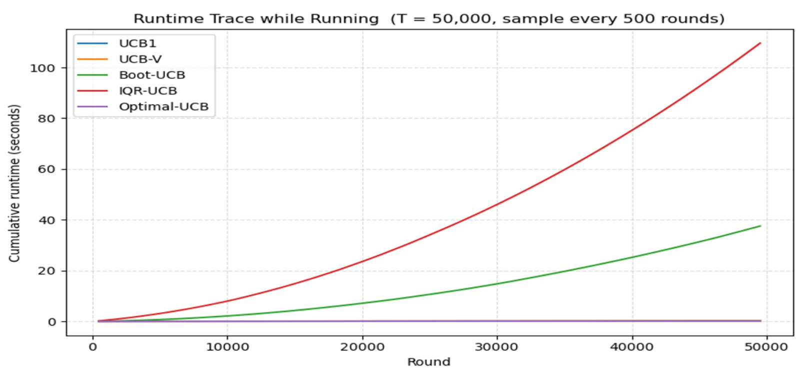

Fig. 3. Runtime Comparison of UCB Variants under the horizon of n = 50000

In the Fig.3, the traces have shown that UCB1, Optimal-UCB and UCB-V all have fast and nearly baseline operation time. However, Bootstrapped UCB and IQR-UCB have significantly longer runtime due to the per-arm resampling and percentile computing every round. In comparison, UCB-V only adds simple variance arithmetic per round, but Bootstrapped UCB rebuilds a bootstrapped sample for every arm each round, with O(t) work per step, and IQR-UCB recomputes quartiles from raw histories per round and is simply heavier.

For online deployment, we will prefer UCB-V and Optimal-UCB for their runtime and regrets. If further robustness is necessary, we need modifications in Bootstrapped-UCB and IQR-UCB to make them more feasible. We can use the runtime trace as a capacity check. If time rises super-linearly, the choice or possible updates may not be preferred.

8 Conclusion

In the Amazon e-commerce stream, the UCB-Variance algorithm achieves the best regret-cost trade-off, with approximately 70% less of cumulative regret than the traditional UCB1 Algorithm and nearly baseline runtime. It flattens cumulative regrets fastest once the arm variances are estimated. Optimal-UCB is a simple and anytime update with 25% less of cumulative regret, as it consistently improves on UCB1 without knowing the horizon. The Bootstrapped UCB, on the contrary, offer little gain: similar cumulative regrets as UCB1, compared to the high computation costs it incurs. The IQR-UCB results in near-zero regret for binary rewards but results from the vanishing bonus instead of a superior learning, exploring too greedily and should be better reserved for value outcomes.

Therefore, the default policy to be deployed should be UCB-Variance algorithm for the heterogeneous categories with different shipping variances. Whenever we want a fast and horizon-free behaviour with minimal code, we could deploy the optimal-UCB algorithm as the low-friction anytime baseline. We should only deploy the IQR-UCB and Bootstrapped UCB for value metrics instead of the binary rewards, and they should be tested further for their robustness to skews and outliers and added safety floors to avoid greedy behaviours if possible.

In the experimental setups, there are still some limitations and possible optimizations to fulfill. For the Bootstrapped UCB, instead of simply computing the bootstrapped mean, we can use an upper bootstrap quantile with bootstrap sample size of 100-300 or adopt the lazy bootstrap that only recomputes the selected arms [6]. This will reduce the exploration time and make the bootstrap mean more robust to the lucky or lucky draws. For the IQR-UCB, we can add a floor that also shrinks with growing sample size to keep a nonzero exploration radius when early quantiles collide and use streaming quantiles to avoid resorting every arm every round. We may also test out the distribution-aware UCB's under these circumstances as they may result in tighter constants than UCB1. For other directions, we may test out the sliding-window UCB for resolving seasonality and product churn.

For further extensions, the evaluation can be improved with multiple objectives. Besides the regrets on return rate reduction, it may extend to kept orders or values metrics of GMV's. The extension may also include the incorporation of contextual bandits using product and user features, evaluating the bounds for warm-start UCB-V algorithm and further analyzing IQR-UCB under different circumstances. In short, UCB-V is the best practical choice for category-level exploration on this dataset, while optimal-UCB is a strong anytime baseline. The robustness-aware UCB variants will become more competitive once taking into considerations of lazy updates and proper confidence schedules. The future works will push these improvements towards possibly value-aware and constrained bandit settings for production scale.

Reference

1. E. Gustafsson, P. Jonsson, J. Holmström, Reducing retail supply chain costs of product returns using digital product fitting. International Journal of Physical Distribution & Logistics Management 51, 877–896 (2021).
2. S. Wang, J.-H. Cheah, X.-J. Lim, Online Shopping Cart Abandonment: A Review and Research Agenda. International Journal of Consumer Studies 47(2), 453–473 (2023).
3. D. Xiang, R. West, J. Wang, X. Cui, J. Huang, Multi-Armed Bandit vs. A/B Tests in E-Commerce: Confidence-Interval & Power Perspectives. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 4204–4214 (2022).
4. P. Auer, N. Cesa-Bianchi, P. Fischer, Finite-Time Analysis of the Multiarmed Bandit Problem. Machine Learning 47(2-3), 235–256 (2002).
5. J.-Y. Audibert, S. Bubeck, C. Munos, Exploration–Exploitation Tradeoff Using Variance Estimates in Multi-Armed Bandits. Theoretical Computer Science 410(19), 1876–1902 (2009).

6. B. Hao, Y. Abbasi-Yadkori, Z. Wen, G. Cheng, Bootstrapping Upper Confidence Bound. In Advances in Neural Information Processing Systems (NeurIPS) 32, 15527–15538 (2019).
7. B. Szörényi, R. Busa-Fekete, P. Weng, E. Hüllermeier, Qualitative Multi-Armed Bandits: A Quantile-Based Approach. In Proceedings of the International Conference on Machine Learning (ICML), 1660–1668 (2015).
8. Y. Liu, L. Li, A Map of Bandits for E-Commerce. (2021).
9. D. Preil, M. Krapp, Bandit-Based Inventory Optimisation: Reinforcement Learning in Multi-Echelon Supply Chains. International Journal of Production Economics 252, 108578 (2022).
10. W. Hoeffding, Probability Inequalities for Sums of Bounded Random Variables. Journal of the American Statistical Association 58(301), 13–30 (1963).
11. T. Lattimore, C. Szepesvári, Bandit Algorithms. Cambridge University Press (2020).